REMARKS

Claims 1-4, 6, 7, 9-17, 19, 20, 22-28, 30, 31 and 33-35 are pending in the present application. By this Response, claims 1, 10, 12-14, 23, 25 and 34 are amended. Claims 1, 12, 14 and 25 are amended to recite "responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state, wherein the detecting step occurs in a device driver and placing step occurs in a firmware." Claims 10, 13, 23 and 34 are amended to recite "responsive to the error counter indicating a number of attempts to recover from the error exceeding a threshold, placing said at least one slot into an unavailable state, wherein the determining step occurs in a device driver and placing step occurs in a firmware." Reconsideration of the claims in view of the above amendments and the following remarks is respectfully requested.

I. <u>35 U.S.C. § 103, Alleged Obviousness, Claims 1, 3, 4, 9, 12, 14, 16, 17, 22, 25, 27, 28 and 33</u>

The Office Action rejects claims 1, 3, 4, 9, 12, 14, 16, 17, 22, 25, 27, 28 and 33 under 35 U.S.C. § 103(a), as being allegedly unpatentable over Seon (U.S. Patent No. 6,574,755 B1) in further view of Adams (U.S. Patent No. 5,379,414). This rejection is respectfully traversed.

As to claims 1, 14 and 25, the Office Action states:

> Referring to claims 1, 14, and 25, Seon discloses responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt (From the abstract, " If the fault occurs on the SCSI bus while the SCSI command is transferred to the target device, the initiator device retries the transfer of the SCSI command to the target device a predetermined number of times."); and responsive to the error exceeding a threshold, placing the hardware component in a permanently unavailable state (From figure 3, element 305. Wherein Seon, at least, places a hardware component in a "permanently" unavailable state by performing a SCSI bus and/or MPU reset.). Although Seon does not specifically disclose the detecting step occurs in a device driver and placing step occurs in a firmware, using a device driver to respond to errors is known in the art and performing

> operations using firmware is notoriously well known in the art. From the abstract of Adams, "A system and method which provides a complete software implementation of a device driver that is capable of detecting an undetectable data corruption problem without hardware redesign and/or internal modification to an existing FDC." A person of ordinary skill in the art at the time of the invention would have been motivated to detect an error with a driver because, from the field of invention from Adams, "eliminates the need for hardware redesign and/or fabrication of new FDCs". Further, Examiner takes official notice for the use of firmware to implement this functionality. An example of this is a BIOS (basic input/output system). A person of ordinary skill in the art at the time of the invention would have been motivated to use a BIOS because it is responsible for the basic input and output functionality, provides an interface between the operating system and system hardware, and supports peripheral technologies and internal services.

Office Action dated July 22, 2004, pages 2-3.

Claim 1, which is representative of the other rejected independent claims 12, 14 and 25 with regard to similarly recited subject matter, reads as follows:

> 1. A method in a data processing system for isolating failing hardware in the data processing system, the method comprising:
>
> responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt; and
>
> responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state, wherein the detecting step occurs in a device driver and placing step occurs in a firmware.

Seon and Adams, taken alone or in combination, fail to teach or suggest responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt, and responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state, wherein the detecting step occurs in a device driver and placing step occurs in a firmware.

Seon is directed to processing a SCSI bus fault in a SCSI system which has an initiator device and a target device interconnected via a SCSI bus. In response to a control command, the Seon system's initiator device requests the target device to execute a specific SCSI command. Then, the initiator device processes a normal script phase in response to the SCSI command to transfer the SCSI command to the target device over the SCSI bus. If the fault occurs on the SCSI bus while the SCSI command is transferred to the target device, the initiator device retries the transfer of the SCSI command to the target device a predetermined number of times.

Thus, with the system of Seon, if an error is detected while the initiator device tries to transfer command to a target device, the initiator device tries to transfer the command another predetermined number of times. In contradistinction, the presently claimed invention detects <u>a recovery attempt from an error</u> for an operation involving a hardware component and stores an indication of the recovery attempt. Seon and Adams, taken alone or in combination, do not teach or suggest this feature. The Office Action alleges that this feature is taught in the Seon abstract, which reads as follows:

> A method for processing a SCSI bus fault in a SCSI system which has an initiator device and a target device interconnected via a SCSI bus. In response to a control command, the initiator device requests the target device to execute a specific SCSI command. Then, the initiator device processes a normal script phase in response to the SCSI command to transfer the SCSI command to the target device over the SCSI bus. If the fault occurs on the SCSI bus while the SCSI command is transferred to the target device, the initiator device retries the transfer of the SCSI command to the target device a predetermined number of times. Therefore, data damage in the target device due to the SCSI bus fault can be reduced to a minimum.

In this section, Seon merely describes <u>detecting an error</u> and if there is an error retrying a transfer of the command from the initiator device to the target device a predetermined number of times. Seon does not monitor for the recovery attempt of device within the system that caused the error on the SCSI bus, but, rather, merely detects the error and tries to transfer the command again. Seon clearly shows the types of faults that are monitored at column 4, lines 37-53, which reads as follows:

> Types of faults occurring on a SCSI bus are as follows:
>
> (1) The first fault type is a SCSI bus busy state where the SCSI bus is timed out because an MPU cannot be synchronized with an acknowledge signal due to instability of a request signal to a target device. The SCSI bus busy state can be recognized when there is no response for 10 seconds after an operation associated with a specific SCSI command is normally completed with no abnormal error by the MPU.
>
> (2) The second fault type is an unexpected disconnect state of the target device where the SCSI target device acts to release the SCSI bus control to enhance performance of the MPU.

(3) The third fault type is a device not ready state where access is attempted when the target device is not ready.

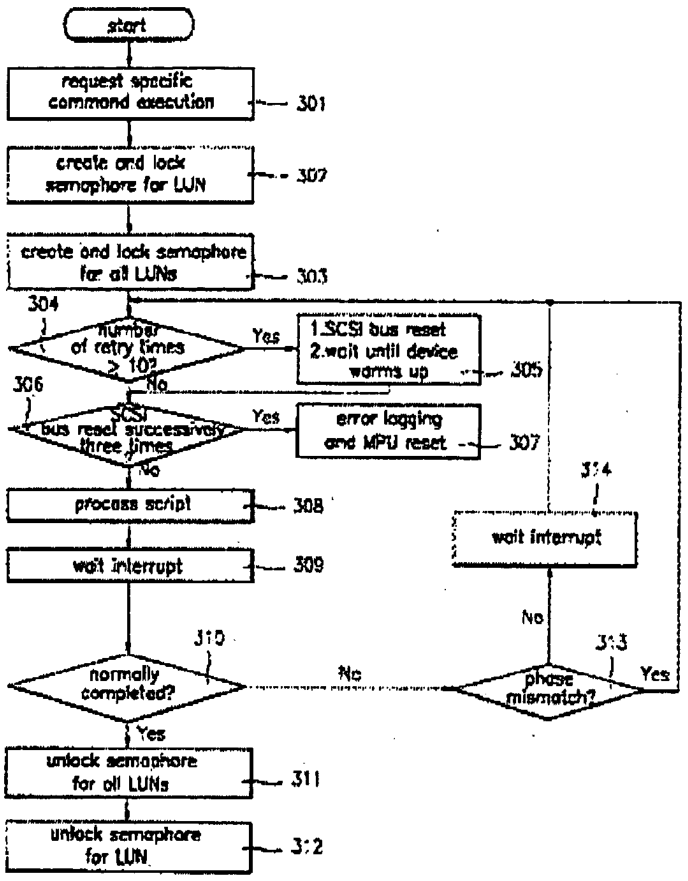
(4) The fourth fault type is a unit attention state where access is attempted just after the target device is reset.

The types of faults detected by Seon is a SCSI bus busy state where the SCSI bus is timed out, a unexpected disconnect state where the target device releases the SCSI bus control, a device not ready state, and a unit attention state where an attempt to transfer command to the target device is not possible because the target device has been reset. None of these detected faults is <u>a recovery attempt from an error</u> for an operation involving a hardware component.

Additionally, Seon does not store an indication of the recovery attempt. The Office Action alleges that this feature is taught in the Seon abstract, shown above. As discussed above, Seon merely stores the number of times the transfer of control from the initiator device to the target device has been tried. The Office Action proffers no explanation as to why storing the number of times a transfer has been attempted is somehow equivalent to storing an indication of a recovery attempt from an error. While Adams may teach software implementation of device drivers, Adams does not make up for the deficiencies of Seon. That is, Adams does not teach or suggest detecting <u>a recovery attempt from an error</u> for an operation involving a hardware component and stores an indication of the recovery attempt. Thus, Seon and Adams, taken alone or in combination, fail to teach or suggest responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt.

Furthermore, Seon and Adams, taken alone or in combination, fail to teach or suggest responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state. The Office Action alleges that this feature is taught by Seon in Figure 3, element 305, which is shown as follows:

FIG.3


start
request specific command execution
301
create and lock semaphore for LUN
302
create and lock semaphore for all LUNs
303
304
number of retry times ≥ 10?
Yes
No
1.SCSI bus reset
2.wait until device warms up
305
306
SCSI bus reset successively three times
Yes
No
error logging and MPU reset
307
process script
308
wait interrupt
309
314
wait interrupt
No
310
313
normally completed?
No
phase mismatch?
Yes
Yes
unlock semaphore for all LUNs
311
unlock semaphore for LUN
312

Seon describes Element 305 as where, if a fault occurs on the SCSI bus successively ten times or more, the first CPU resets the SCSI bus and initializes a SCSI register in the SCSI card. After the SCSI resets, the first CPU waits until the RAID controller warms up and then retries transferring the SCSI command to the target device. Thus, Seon merely resets the SCSI bus, rather than placing the device that has met a threshold, which indicates the number of attempts where the device has attempted recovery, in an unavailable state. Seon does not intend to place any device in an unavailable state as after the SCSI is reset, the transfer of the SCSI command to the target device is retried (Seon, column 5, lines 30-33). Again, Adams does not make up for the deficiencies of Seon. That is, Adams does not teach or suggest responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state. Thus, Seon and Adams, taken alone or in combination, fail to teach or suggest responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state.

Additionally, Seon and Adams, taken alone or in combination, fail to teach responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state, <u>wherein the detecting step occurs in a device driver and placing step occurs in a firmware</u>. The Office Action admits that Seon does not teach or suggest detecting a recovery attempt from an error for an operation involving a hardware component in a device driver and placing the hardware component in an unavailable state in firmware. The Office Action further alleges "using a device driver to respond to errors is known in the art and performing operations using firmware is notoriously well known in the art." However, as discussed above, Applicants are not merely claiming detecting errors, but, rather, detecting <u>a recovery attempt from an error</u> for an operation involving a hardware component using a device driver. Applicants respectfully submit that detecting <u>a recovery attempt from an error</u> for an operation involving a hardware component using a device driver was not well known at the time of the invention.

Further, the Office Action alleges that Adams teaches detecting a recovery attempt from an error for an operation involving a hardware component in a device driver in the Adams abstract, which reads as follows:

> A system and method which provides a complete software implementation of a device driver that is capable of detecting an undetectable data corruption problem without hardware redesign and/or internal modification to an existing FDC. The approach taken consists of software DMA shadowing and use of a software decoding network which allows the implementation of the invention to require a small amount of memory and only degrade the performance of the computer system a minimal amount when floppy diskette write operations occur.

In this section, Adams merely describes detecting errors using a device driver. Adams does not make up for the deficiencies of Seon. Adams is directed to providing a device driver that detects an undetectable data corruption problem. Nowhere in this section, or any other section of Seon or Adams, is it taught or suggested to detecting a recovery attempt from an error for an operation involving a hardware component using a device driver. Moreover, Adams is directed to detecting data corruption errors and not a recovery attempt of a hardware component. Thus, Applicants respectfully submit that Seon and Adams, taken alone or in combination, fail to teach or suggest detecting a recovery attempt from an error for an operation involving a hardware component using a device driver.

Still further, Applicants respectfully traverse the Examiner's Official Notice for the use of firmware to implement placing the hardware component in an unavailable state. Applicants respectfully submit that Seon and Adams, taken alone or in combination, fail to teach or suggest placing the hardware component, using firmware, in an unavailable state in responsive to a number of attempts to recover from the error exceeding a threshold. Since neither Seon nor Adams teach or suggest this feature, Applicants respectfully request that the Examiner provide a reference that teaches or suggests placing a hardware component, using firmware, in an unavailable state in responsive to a number of attempts to recover from the error exceeding a threshold.

Furthermore, there is not so much as a suggestion in either reference to modify the references to include such features. That is, there is no teaching or suggestion in Seon or Adams that a problem exists for which storing an indication of an recovery attempt in response to detecting a recovery attempt from an error for an operation involving a hardware component and placing the hardware component in an unavailable state in response to a number of attempts to recover from the error exceeding a threshold, wherein the detecting step occurs in a device driver and placing step occurs in a firmware, is a solution. To the contrary, Seon teaches detecting an error and, if there is an error, retrying a transfer of the command from the initiator device to the target device a predetermined number of times. Adams teaches a device driver that detects an undetectable data corruption problem. Neither reference even recognizes a need to store an indication of a recovery attempt in response to detecting a recovery attempt from an error for an operation involving a hardware component and place the hardware component in an unavailable state in response to a number of attempts to recover from the error exceeding a threshold, as recited in claim 1.

Moreover, neither reference teaches or suggests the desirability of incorporating the subject matter of the other reference. That is, there is no motivation offered in either reference for the alleged combination. The Office Action alleges that the motivation for the combination is because Adams "eliminates the need to hardware redesign and/or fabrication of new FDCs." As discussed above, Adams is directed to detecting data corruption errors and not a recovery attempt of a hardware component. Neither reference detects a recovery attempt from an error for an operation involving a hardware component. Thus, the only teaching or suggestion to even attempt the alleged combination is based on a prior knowledge of Applicants' claimed invention thereby constituting impermissible hindsight reconstruction using Applicants' own disclosure as a guide.

One of ordinary skill in the art, being presented only with Seon and Adams, and without having a prior knowledge of Applicants' claimed invention, would not have found it obvious to combine and modify Seon and Adams to arrive at Applicants' claimed invention. To the contrary, even if one were somehow motivated to combine Seon and Adams, and it were somehow possible to combine the two systems, the result would not be the invention, as recited in claim 1. The result would be simply detecting errors and in response retrying the command that encountered the error. The resulting system still would not store an indication of a recovery attempt in response to detecting a recovery attempt from an error for an operation involving a hardware component and place the hardware component in an unavailable state in response to a number of attempts to recover from the error exceeding a threshold.

Thus, Seon and Adams, taken alone or in combination, fail to teach or suggest all of the features in independent claims 1, 12, 14 and 25. At least by virtue of their dependency on claims 1, 14 and 25, the specific features of claims 3, 4, 9, 16, 17, 22, 27, 28 and 33 are not taught or suggested by Seon and Adams, either alone or in combination. Accordingly, Applicants respectively requests withdrawal of the rejection of claims 1, 3, 4, 9, 12, 14, 16, 17, 22, 25, 27, 28 and 33 under 35 U.S.C. § 103(a).

Moreover, in addition to their dependency from independent claims 1, 14 and 25 respectively, the specific features recited in dependent claims 3, 4, 9, 16, 17, 22, 27, 28 and 33 are not taught or suggested by Seon and Adams, taken alone or in combination. For example, with regard to claims 3, 16 and 27, neither Seon nor Adams, taken alone or in combination, fairly teaches or suggests making a call to a hardware interface layer to place the hardware component into a permanent reset state. The Office Action alleges that this feature is taught by Seon in Figure 3, Element 307, shown above. Element 307 is described by Seon as an MPU reset upon determining that the SCSI bus 51 has been reset successively three times. Thus, upon detecting an error on the SCSI bus which causes the SCSI bus to reset three times, Seon resets the MPU. Resetting the MPU is not placing the hardware component, in the Seon case the SCSI bus, in an unavailable state responsive to a number of attempts to recover from the error exceeding a threshold. Adams does not make up for the deficiencies of Seon, as Adams fails to teach or suggest making a call to a hardware interface layer to place the hardware component into a permanent reset state.

Thus, in addition to being dependent on independent claims 1, 14 and 25, the specific features of dependent claims 3, 4, 9, 16, 17, 22, 27, 28 and 33 are also distinguishable over Seon and Adams by virtue of the specific features recited in these claims. Accordingly, Applicant respectfully requests withdrawal of the rejection of dependent claims 3, 4, 9, 16, 17, 22, 27, 28 and 33 under 35 U.S.C. § 103 (a).

II. <u>35 U.S.C. § 103, Alleged Obviousness, Claims 1, 3, 4, 9-14, 16, 17, 22-25, 27, 28, 33, 34 and 35</u>

The Office Action rejects claims 1, 3, 4, 9-14, 16, 17, 22-25, 27, 28 and 33-35 under 35 U.S.C. § 103(a), as being allegedly unpatentable by Oberhauser (U.S. Patent No. 6,711,702 B1) in further view of Adams (U.S. Patent No. 5,379,414). This rejection is respectfully traversed.

As to claims 1, 14 and 25, the Office Action states:

> Referring to claims 1, 14, and 25, Oberhauser discloses responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt; and responsive to the error exceeding a threshold, placing the hardware component in a permanently unavailable state (From the abstract, "A repetition counter for counting a number of start-up attempts is provided for a restarting procedure. During a locked phase, the peripheral unit that is affected is temporarily taken out of service. After that, a monitoring phase with a temporary start-up is initiated during which tests for faults are carried out. If the unit is determined to be free from faults, a final start-up takes place

> following the monitoring phase. In the case of a fault during the monitoring phase, the count of the repetition counter is compared with a threshold value. A final taking-out-of-service takes place if the count of the repetition counter exceeds the threshold value.") Although Oberhauser does not specifically disclose the detecting step occurs in a device driver and placing step occurs in a firmware, using a device driver to respond to errors is known in the art and performing operations using firmware is notoriously well known in the art. From the abstract of Adams, "A system and method which provides a complete software implementation of a device driver that is capable of detecting an undetectable data corruption problem without hardware redesign and/or internal modification to an existing FDC." A person of ordinary skill in the art at the time of the invention would have been motivated to detect an error with a driver because, from the field of invention from Adams, "eliminates the need for hardware redesign and/or fabrication of new FDCs". Further, Examiner takes official notice for the use of firmware to implement this functionality. An example of this is a BIOS (basic input/output system). A person of ordinary skill in the art at thet ime of the invention would have been motivated to use a BIOS because it is responsible for the basic input and output functionality, provides an interface between the operating system and system hardware, and supports peripheral technologies and internal services.

Office Action dated July 22, 2004, pages 4-5.

Oberhauser and Adams, taken alone or in combination, fail to teach or suggest responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt, and responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state, wherein the detecting step occurs in a device driver and placing step occurs in a firmware.

Oberhauser is directed to a method that defines steps and sequences for dealing with peripheral units reported as defective in a communications system. Oberhauser provides a repetition counter for counting a number of start-up attempts for a restarting procedure. During a locked phase, the peripheral unit that is affected is temporarily taken out of service. After that, a monitoring phase with a temporary start-up is initiated during which tests for faults are carried out. If the unit is determined to be free from faults, a final start-up takes place following the monitoring phase. In the case of a fault during the monitoring phase, the count of the repetition counter is compared with a threshold value. A final taking-out-of-service takes place if the count of the repetition counter exceeds the threshold value. Otherwise, the repetition counter is incremented and another transition into the locked phase takes place. The duration of the locked phase is dependent on the count of the repetition counter.

Thus, with the system of Oberhauser, if a peripheral unit is reported as defective, the unit is locked-out and a temporary start-up is initiated to test for faults. If a fault is detected, a counter is incremented and another restart is initiated. Once the counter reaches a threshold, the unit is taken out of service. In contradistinction, the presently claimed invention detects a recovery attempt from an error for an operation involving a hardware component in normal operation and stores an indication of the recovery attempt. Oberhauser and Adams, taken alone or in combination, do not teach or suggest this feature. The Office Action alleges that this feature is taught in the Oberhauser abstract, which reads as follows:

> The method defines steps and sequences for dealing with peripheral units reported as defective in a communications system. A repetition counter for counting a number of start-up attempts is provided for a restarting procedure. During a locked phase, the peripheral unit that is affected is temporarily taken out of service. After that, a monitoring phase with a temporary start-up is initiated during which tests for faults are carried out. If the unit is determined to be free from faults, a final start-up takes place following the monitoring phase. In the case of a fault during the monitoring phase, the count of the repetition counter is compared with a threshold value. A final taking-out-of-service takes place if the count of the repetition counter exceeds the threshold value. Otherwise, the repetition counter is incremented and another transition into the locked phase takes place. The duration of the locked phase is dependent on the count of the repetition counter.

In this section, Oberhauser merely describes testing a peripheral unit, which has been reported as defective and has been locked-out by initiating a temporary start-up. If a fault is detected during the temporary start-up, a counter is incremented indicating the number of restarts and another restart is initiated. Once the counter reaches a threshold, the unit is taken out of service. Oberhauser does not monitor during normal operation for the recovery attempt of a device within the unit that caused the error, but, rather, merely takes note of the error and initiates another restart. Thus, if a unit within the Oberhauser system does not start-up within a certain number of start-up attempts, the unit is placed into an out-of-service condition. Thus, Oberhauser and Adams, taken alone or in combination, fail to teach or suggest responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt.

Furthermore, Oberhauser and Adams, taken alone or in combination, fail to teach or suggest responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state. The Office Action alleges that this feature is taught in the Oberhauser abstract, shown above. As discussed above, if a unit within the Oberhauser system does not start-up within a certain number of start-up attempts, the unit is placed into an out-of-service condition. This process is not equivalent to responsive <u>to a number of attempts to recover from the error exceeding a threshold</u>, placing the hardware component in an unavailable state. Again, Adams does not make up for the deficiencies of Oberhauser. That is, Adams does not teach or suggest responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state. Thus, Oberhauser and Adams, taken alone or in combination, fail to teach or suggest responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state.

Additionally, Oberhauser and Adams, taken alone or in combination, fail to teach responsive to a number of attempts to recover from the error exceeding a threshold, placing the hardware component in an unavailable state, <u>wherein the detecting step occurs in a device driver and placing step occurs in a firmware</u>. The Office Action admits that Oberhauser does not teach or suggest detecting a recovery attempt from an error for an operation involving a hardware component in a device driver and placing the hardware component in an unavailable state in firmware. The Office Action further alleges "using a device driver to respond to errors is known in the art and performing operations using firmware is notoriously well known in the art." However, as discussed above, Applicants are not merely claiming detecting errors, but, rather, detecting <u>a recovery attempt from an error</u> for an operation involving a hardware component using a device driver. Applicants respectfully submit that detecting <u>a recovery attempt from an error</u> for an operation involving a hardware component using a device driver is not well known at the time of the invention.

Further, the Office Action alleges that Adams teaches detecting a recovery attempt from an error for an operation involving a hardware component in a device driver in the Adams abstract, which reads as follows:

> A system and method which provides a complete software implementation of a device driver that is capable of detecting an undetectable data corruption problem without hardware redesign and/or internal modification to an existing FDC. The approach taken consists of software DMA shadowing and use of a software decoding network which allows the implementation of the invention to require a small amount of memory and only degrade the performance of the computer system a minimal amount when floppy diskette write operations occur.

In this section, Adams merely describes detecting errors using a device driver. Adams does not make up for the deficiencies of Oberhauser. Adams is directed to providing a device driver that detects an undetectable data corruption problem. Nowhere in this section, or any other section of Oberhauser or Adams, is it taught or suggested to detecting a recovery attempt from an error for an operation involving a hardware component using a device driver. Moreover, Adams is directed to detecting data corruption errors and not a recovery attempt of a hardware component. Thus, Applicants respectfully submit that Oberhauser and Adams, taken alone or in combination, fail to teach or suggest detecting a recovery attempt from an error for an operation involving a hardware component using a device driver.

Still further, Applicants respectfully traverse the Examiner's Official Notice for the use of firmware to implement placing the hardware component in an unavailable state. Applicants respectfully submit that Oberhauser and Adams, taken alone or in combination, fail to teach or suggest placing the hardware component, using firmware, in an unavailable state in responsive to a number of attempts to recover from the error exceeding a threshold. Since neither Oberhauser nor Adams teach or suggest this feature, Applicants respectfully request that the Examiner provide a reference that teaches or suggests placing a hardware component, using firmware, in an unavailable state in responsive to a number of attempts to recover from the error exceeding a threshold.

As to claims 10, 23 and 34, the Office Action states:

> Referring to claims 10, 23, and 34, Oberhauser discloses a method in a data processing system for handling errors, the method comprising: responsive to an occurrence of an error, determining whether the error is a

> recoverable error (From the abstract, "A repetition counter for counting a number of start-up attempts is provided for a restarting procedure. During a locked phase, the peripheral unit that is affected is temporarily taken out of service. After that, a monitoring phase with a temporary start-up is initiated during which tests for faults are carried out. If the unit is determined to be free from faults, a final start-up takes place following the monitoring phase. In the case of a fault during the monitoring phase, the count of the repetition counter is compared with a threshold value."); responsive to a determination that the error is a recoverable error, identifying at least one slot on a bus indicating an error state (From the abstract, "After that, a monitoring phase with a temporary start-up is initiated during which tests for faults are carried out." Further, from line 23 of column 2, "In accordance with an additional feature of the invention, a plurality of peripheral units are allocated the following hierarchy levels: a) line unit; b) assembly; c) circuit; d) terminal."; incrementing an error counter for said identified at least one identified slot (From the abstract, "A repetition counter for counting a number of start-up attempts is provided for a restarting procedure."); and responsive to the error counter exceeding a threshold, placing said at least one slot into an unavailable state (From the abstract, "A final taking-out-of-service takes place if the count of the repetition counter exceeds the threshold value.").

Office Action dated July 22, 2004, pages 6-7.

Claim 10, which is representative of the other rejected independent claims 13, 23 and 34 with regard to similarly recited subject matter, reads as follows:

> 10. A method in a data processing system for handling errors, the method comprising:
>
> responsive to an occurrence of an error, determining whether the error is a recoverable error;
>
> responsive to a determination that the error is a recoverable error, identifying at least one slot on a bus indicating an error state;
>
> incrementing an error counter for said at least one identified slot; and
>
> responsive to the error counter indicating a number of attempts to recover from the error exceeding a threshold, placing said at least one slot into an unavailable state, wherein the determining step occurs in a device driver and placing step occurs in a firmware.

Oberhauser and Adams, taken alone or in combination, fail to teach or suggest responsive to an occurrence of an error, determining whether the error is a recoverable error, responsive to a determination that the error is a recoverable error, identifying at least one slot on a bus indicating an error state and responsive to the error counter indicating a number of attempts to recover from the error exceeding a threshold, placing said at least one slot into an unavailable state, wherein the determining step occurs in a device driver and placing step occurs in a firmware.

The Oberhauser system tests a peripheral unit that is reported as defective for faults by locking-out the unit and initiating a temporary start-up. If a fault is detected, a counter is incremented and another restart is initiated. Once the counter reaches a threshold, the unit is taken out of service. In contradistinction, the presently claimed invention determines in normal operation whether the error is a recoverable error responsive to an occurrence of an error, if the error is recoverable identifies the slot on a bus indicating an error state and responsive to an error counter indicating a number of attempts to recover from the error exceeding a threshold, placing the slot into an unavailable state. Oberhauser does not determine whether the error that occurs on start up of the peripheral unit is a recoverable error. The Oberhauser system merely increments a counter if any error is encountered. Additionally, the reference does not identify a slot on a bus indicating the error state. Oberhauser is monitoring a peripheral unit. Further, Oberhauser teaches places the unit into an out-of-service state if the number of start-up errors exceeds a threshold. This teaching of Oberhauser does not teach or suggest responsive to an error counter indicating a number of attempts to recover from the error exceeding a threshold, placing the slot into an unavailable state. Adams does not make up for the deficiencies of Oberhauser, as Adams does not teach or suggest these specific features. Thus, Oberhauser and Adams, taken alone or in combination, fail to teach or suggest responsive to detecting a recovery attempt from an error for an operation involving a hardware component, storing an indication of the attempt.

Furthermore, there is not so much as a suggestion in either reference to modify the references to include such features. That is, there is no teaching or suggestion in Oberhauser or Adams that a problem exists for which storing an indication of an recovery attempt in response to detecting a recovery attempt from an error for an operation involving a hardware component and placing the hardware component in an unavailable state in response to the number of attempts to recover from the error exceeding a threshold, wherein the detecting step occurs in a device driver and placing step occurs in a firmware, is a solution. To the contrary, Oberhauser teaches detecting an error and if there is an error retrying a transfer of the command from the initiator device to the target device a predetermined number of times. Adams teaches a device driver that detects an undetectable data corruption problem. Neither reference even recognizes a need to store an indication of a recovery attempt in response to detecting a recovery attempt from an error for an operation involving a hardware component and place the hardware component in an unavailable state in response to the number of attempts to recover from the error exceeding a threshold, as recited in claim 1.

Moreover, neither reference teaches or suggests the desirability of incorporating the subject matter of the other reference. That is, there is no motivation offered in either reference for the alleged combination. The Office Action alleges that the motivation for the combination is because Adams "eliminates the need to hardware redesign and/or fabrication of new FDCs." As discussed above, Adams is directed to detecting data corruption errors and not a recovery attempt of a hardware component. Neither reference detects a recovery attempt from an error for an operation involving a hardware component. Thus, the only teaching or suggestion to even attempt the alleged combination is based on a prior knowledge of Applicants' claimed invention thereby constituting impermissible hindsight reconstruction using Applicants' own disclosure as a guide.

One of ordinary skill in the art, being presented only with Oberhauser and Adams, and without having a prior knowledge of Applicants' claimed invention, would not have found it obvious to combine and modify Oberhauser and Adams to arrive at Applicants' claimed invention. To the contrary, even if one were somehow motivated to combine Oberhauser and Adams, and it were somehow possible to combine the two systems, the result would not be the invention, as recited in claim 1. The result would be simply detecting errors and in response retrying the command that encountered the error. The resulting system still would not store an indication of a recovery attempt in response to detecting a recovery attempt from an error for an operation involving a hardware component and place the hardware component in an unavailable state in response to the number of attempts to recover from the error exceeding a threshold.

Thus, Oberhauser and Adams, taken alone or in combination, fail to teach or suggest all of the features in independent claims 1, 10, 12-14, 23, 25 and 34. At least by virtue of their dependency on claims 1, 10, 14, 23, 25 and 34, the specific features of claims 3, 4, 9, 11, 16, 17, 22, 24, 27, 28, 33 and 35 are not taught or suggested by Oberhauser and Adams, either alone or in combination. Accordingly, Applicants respectively requests withdrawal of the rejection of claims 1, 3, 4, 9-14, 16, 17, 22-25, 27, 28 and 33-35 under 35 U.S.C. § 103(a).

Moreover, in addition to their dependency from independent claims 1, 10, 14, 23, 25 and 34 respectively, the specific features recited in dependent claims 3, 4, 9, 11, 16, 17, 22, 24, 27, 28, 33 and 35 are not taught or suggested by Oberhauser and Adams, taken alone or in combination. For example, with regard to claims 3, 16 and 27, neither Oberhauser nor Adams, taken alone or in combination, fairly teaches or suggests making a call to a hardware interface layer to place the hardware component into a permanent reset state. The Office Action alleges that this feature is taught by Oberhauser at column 3, lines 39-40, which reads as follows:

If the result is negative, a final decommissioning OFF_SERV takes place.

In this section, Oberhauser places a unit in an out-of-service state if the unit exceeds a predetermined number of start-up errors. Applicants respectfully submit that placing a unit in an out-of-service state is not the same as placing the unit in a permanent reset state. Adams does not make up for the deficiencies of Oberhauser, as Adams fails to teach or suggest making a call to a hardware interface layer to place the hardware component into a permanent reset state.

Thus, in addition to being dependent on independent claims 1, 14 and 25, the specific features of dependent claims 3, 4, 9, 11, 16, 17, 22, 24, 27, 28, 33 and 35 are also distinguishable over Oberhauser and Adams by virtue of the specific features recited in these claims. Accordingly, Applicant respectfully requests withdrawal of the rejection of dependent claims 3, 4, 9, 11, 16, 17, 22, 24, 27, 28, 33 and 35 under 35 U.S.C. § 103 (a).

III. <u>35 U.S.C. § 103, Alleged Obviousness, Claims 2, 15 and 26</u>

The Office Action rejects claims 2, 15 and 26 under 35 U.S.C. § 103(a), as being allegedly unpatentable by Seon (U.S. Patent No. 6,574,755 B1) in view of Adams (U.S. Patent No. 5,379,414) as applied to claims 1, 14 and 25 above, and further in view of Chen et al. (U.S. Patent No. 6,591,324 B1). This rejection is respectfully traversed.

Claims 2, 15 and 26 are dependent on independent claims 1, 14 and 25 and, thus, these claims distinguish over Seon and Adams for at least the reasons noted above with regards to claims 1, 14 and 25. Moreover, Chen does not provide for the deficiencies of Seon and Adams, and, thus, any alleged combination of Seon, Adams and Chen would not be sufficient to reject independent claims 1, 14 and 25 or claims 2, 15 and 26 by virtue of their dependency.

Moreover, the Office Action may not use the claimed invention as an "instruction manual" or "template" to piece together the teachings of the prior art so that the invention is rendered obvious. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Such reliance is an impermissible use of hindsight with the benefit of Applicant's disclosure. *Id.* Therefore, absent some teaching, suggestion, or incentive in the prior art, Seon and Chen cannot be properly combined to form the claimed invention. As a result, absent any teaching, suggestion, or incentive from the prior art to make the proposed combination, the presently claimed invention can be reached only through an impermissible use of hindsight with the benefit of Applicant's disclosure a model for the needed changes.

In view of the above, Seon, Adams and Chen, taken either alone or in combination, fail to teach or suggest the specific features recited in independent claims 1, 14 and 25, from which claims 2, 15 and 26 depend. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 2, 15 and 26 under 35 U.S.C. § 103.

IV. <u>35 U.S.C. § 103, Alleged Obviousness, Claims 2, 15 and 26</u>

The Office Action rejects claims 2, 15 and 26 under 35 U.S.C. § 103(a), as being allegedly unpatentable by Oberhauser (U.S. Patent No. 6,711,702 B1) in view of Adams (U.S. Patent No. 5,379,414) as applied to claims 1, 14 and 25 above, and further in view of Chen et al. (U.S. Patent No. 6,591,324 B1). This rejection is respectfully traversed.

Claims 2, 15 and 26 are dependent on independent claims 1, 14 and 25 and, thus, these claims distinguish over Oberhauser and Adams for at least the reasons noted above with regards to claims 1, 14 and 25. Moreover, Chen does not provide for the deficiencies of Oberhauser and Adams, and, thus, any alleged combination of

Oberhauser, Adams and Chen would not be sufficient to reject independent claims 1, 14 and 25 or claims 2, 15 and 26 by virtue of their dependency.

Moreover, the Office Action may not use the claimed invention as an "instruction manual" or "template" to piece together the teachings of the prior art so that the invention is rendered obvious. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Such reliance is an impermissible use of hindsight with the benefit of Applicant's disclosure. *Id.* Therefore, absent some teaching, suggestion, or incentive in the prior art, Oberhauser, Adams and Chen cannot be properly combined to form the claimed invention. As a result, absent any teaching, suggestion, or incentive from the prior art to make the proposed combination, the presently claimed invention can be reached only through an impermissible use of hindsight with the benefit of Applicant's disclosure a model for the needed changes.

In view of the above, Oberhauser, Adams and Chen, taken either alone or in combination, fail to teach or suggest the specific features recited in independent claims 1, 14 and 25, from which claims 2, 15 and 26 depend. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 2, 15 and 26 under 35 U.S.C. § 103.

V. <u>35 U.S.C. § 103, Alleged Obviousness, Claims 3, 6, 7, 16, 19, 20, 27, 30 and 31</u>

The Office Action rejects claims 3, 6, 7, 16, 19, 20, 27, 30 and 31 under 35 U.S.C. § 103(a), as being allegedly unpatentable by Seon (U.S. Patent No. 6,574,755 B1) in view of Adams (U.S. Patent No. 5,379,414) as applied to claims 1, 14 and 25 above. This rejection is respectfully traversed.

Claims 3, 6, 7, 16, 19, 20, 27, 30 and 31 are dependent on independent claims 1, 14 and 25 and, thus, these claims distinguish over Seon and Adams for at least the reasons noted above with regards to claims 1, 14 and 25.

Thus, in view of the above, Seon and Adams, taken alone or in combination, fail to teach or suggest the specific features recited in independent claims 1, 14 and 25, from which claims 3, 6, 7, 16, 19, 20, 27, 30 and 31 depend. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 3, 6, 7, 16, 19, 20, 27, 30 and 31 under 35 U.S.C. § 103.

VI. <u>Conclusion</u>

It is respectfully urged that the subject application is patentable over the prior art of record and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

Respectfully submitted,

DATE: October 18, 2004

Francis Lammes

Francis Lammes
Reg. No. 55,353
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 367-2001
Agent for Applicants